UNIVERSIDAD TECNOLÓGICA NACIONAL

FACULTAD REGIONAL SAN NICOLÁS

INGENIERIA ELECTRÓNICA

TÉCNICAS DIGITALES III

Processor Expert

Manual de Usuario

Sebastián Cerella – UTN-FRSN

AÑO 2009

Processor Expert

Processor Expert es un plug-in del Code Warrior y está diseñado para el rápido desarrollo de aplicaciones para un amplio rango de DSCs. Su función principal es controlar la CPU, los recursos de hardware y permitir el diseño del prototipo de manera virtual.

Sus principales características son:

- Las aplicaciones son creadas partiendo de *componentes* llamados *Embedded Beans*.
- Los Embedded Beans encapsulan la funcionalidad de elementos básicos como CPUs, periféricos on-chip de la CPU, PGA, dispositivos virtuales, etc.
- Processor Expert sugiere, conecta y genera los drivers para el hardware de un sistema embebido, periféricos o los algoritmos usados. Esto permite al usuario concentrarse en la parte creativa del proyecto.
- El usuario comienza diseñando directamente definiendo el comportamiento de la aplicación en lugar de desperdiciar días tratando de hacer funcionar el chip.
- Los beans de inicialización generan el código necesario para setear los dispositivos on-chip y soportar todas sus características.
- Processor Expert permite examinar fácilmente los detalles de la arquitectura y la relación entre el setup de los *beans* y la inicialización de los registros de control de la CPU.
- El usuario puede crear sus propios beans usando el **Bean Wizard**.

El Code Warrior posee un menú con un ítem llamado Processor Expert (PE desde ahora). El PE genera el código desde los beans y el Code Warrior gestiona los archivos del proyecto, la compilación y el proceso de *debugging*.

Bean.
Un *bean* es un objeto con una función definida. Puede ser accedido por *propiedades*, *métodos* y *eventos*. Las *propiedades* definen lo parámetros del bean (inicialización, seteos, características, conexión con la CPU).
Usando los *métodos* se puede setear el estado del bean y leer los resultados del mismo.
Los *eventos* son llamados por el driver del bean cuando ocurre algo relacionado con el evento, por ejemplo, interrupciones por hardware.
El *driver* del bean contiene la implementación de los métodos y las llamadas cuando se dan los eventos del mismo.
Un *software bean* es un bean que no tiene acceso directo al hardware. El acceso al hardware (inicialización y control) puede ser hecho usando herencia de otro bean que si la tenga.
Un *pure software bean* es idéntico al anterior pero sin posibilidad de heredar de ningún *hardware bean*.
El *driver* contiene los códigos fuentes de todos los métodos y eventos del bean. Cada *bean* (excepto el driver de CPU) tiene un driver asociado consigo.
El usuario puede crear un bean. Después de su creación, se tiene que escribir el código de todos los nuevos (no heredados) métodos que haya creado para ese bean. El código se escribe en un macro lenguaje especial.
Ejemplo.

Para crear un proyecto en PE debemos ejecutar el Code Warrior. Luego, seleccionando ***File->New*** se abre la ventana en donde nos solicita indicar que tipo de proyecto deseamos crear.

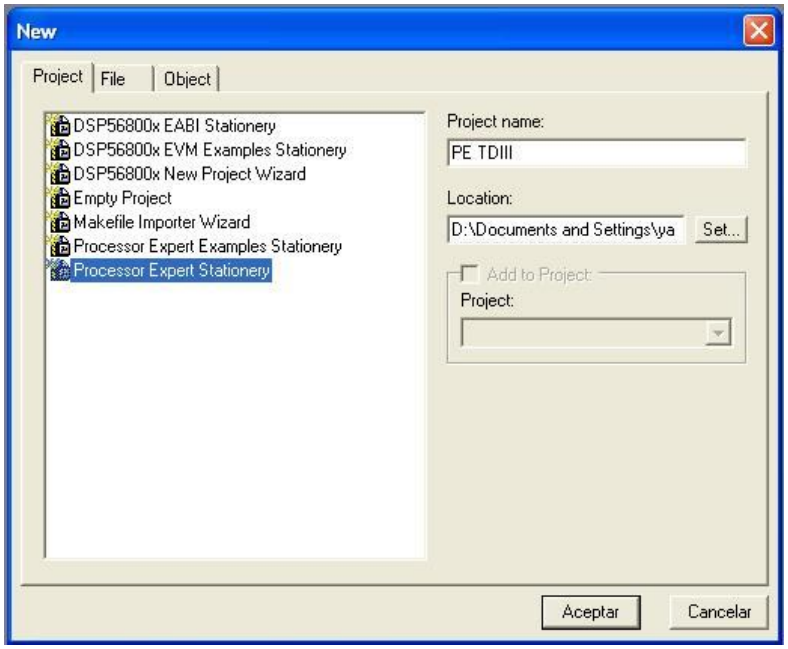

Figura 1

Según se puede observar en la figura anterior se dispone de las siguientes opciones:

- EABI[1] Stationery[2]: Permite crear un proyecto para las placas de Freescale con el estándar EABI para tener compatibilidad con OS principalmente.
- EVM Example Stationery: Permite crear una aplicación simple, a modo de ejemplo, para las placas de evaluación de Freescale. Estos ejemplos forman parte del paquete del Code Warrior. Es ideal como punto de partida para realizar pruebas y hacer los primeros contactos con el IDE y las placas EVM.
- New Proyect Wizard: Es un asistente para la creación de proyectos nuevos que facilita la configuración del dispositivo y carga las librerías estándar.
- Emply Proyect: Es un proyecto totalmente vacío sin definiciones específicas para una placa particular.
- Makefile Importer Wizard: se utiliza para abrir un proyecto basado en makefile a un proyecto manejado por el IDE.
- Processor Expert Examples Stationery: Crea proyectos con ejemplos listos para las placas de Freescale dentro del entorno del Processor Expert.
- Processor Expert Stationary: Sirve para crear proyectos con PE basado en un template (stationary) para una placa particular.

En nuestro caso utilizaremos el PE para generar un proyecto simple que nos facilite el primer contacto con el IDE.

---

[1] El estandard EABI (embedded-application binary interface) especifica convenciones estandares para formatos de archivos, tipos de datos, uso de registros, organización de stacks y los parámetros de funciones pasados a un programa de software embebido. Los compiladores que soportan EABI crean código objeto que es compatible con el código generado por otro compilador. Esto permite, a los desarrolladores linquear con librerías generadas con otros compiladores.

[2] Stationery es un template que incluye un proyecto completo prearmado con archivos fuentes, librerías y el apropiado seteo para el compilador y el linqueador.

En la figura 1 se puede observar el ítem a seleccionar para nuestro caso, *Processor Expert Stationery*. Luego ingresaremos el nombre del proyecto, por ejemplo *PE TDIII* y, de ser necesario, se le dará una nueva ubicación al mismo presionando *Set*. El path que indiquemos allí quedará configurado como el path base del proyecto. Por el momento no utilizaremos las solapas File y Object ya que sirven para incluir archivos fuentes u objeto a nuestro proyecto generados en otro proyecto. Al presionar *Aceptar* se abre la siguiente ventana:

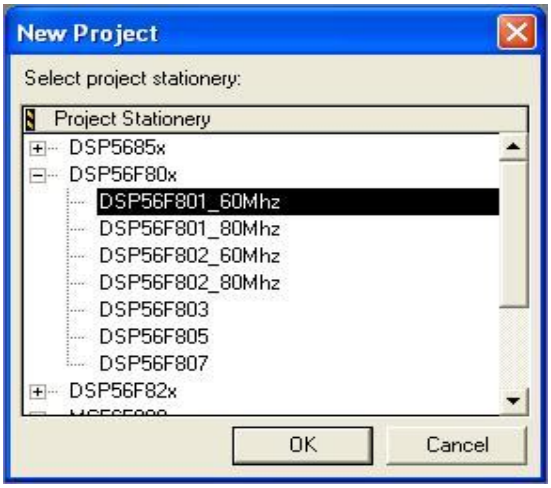

Figura 2

En la figura 2 se puede observar el ítem a seleccionar, *DSP56F80x -> DSP56F801_60Mhz*.
Al confirmar esta selección con *OK* el IDE comienza a generar las configuraciones necesarias según la información que le hemos indicado y abre el PE.

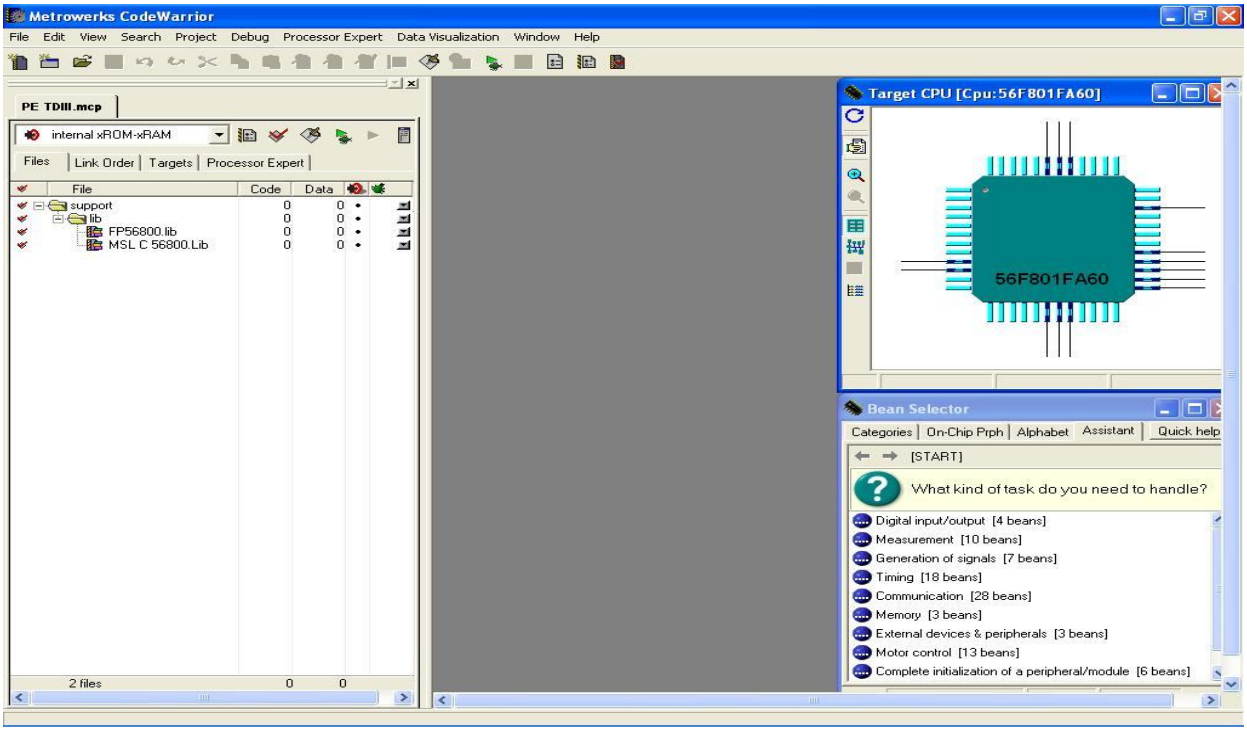

Figura 3

En la figura 3 se observa el IDE con las siguientes ventanas abiertas:

- *Bean Selector*: aquí aparecen todos los *bean* disponibles ordenados por grupos funcionales.
- *Target CPU*: aquí aparece el esquema de la CPU seleccionada. Desde esta ventana se puede acceder a información de los pines del DSC con su respectiva funcionalidad y nro de pin, por ejemplo. Además, muestra un paneo general de los pines que tienen asociado un bean y como está configurado el mismo. Se puede ver que existen pines ya conectados. El PE lo muestra de esta forma porque son pines que no están disponibles para el usuario ya que son de alimentación de la CPU, referencia para el ADC y de capacitares de filtraje para el apropiado funcionamiento de la CPU.
- *Proyect Panel*: es el panel que está sobre la izquierda de la pantalla. Allí aparece todo lo referente a la configuración del proyecto y sus archivos asociados. Por el momento, PE agregó solo las librerías estándar. Cuando compilemos el proyecto se incluirán todos los archivos que hayamos incluido y configurado.

En nuestro caso intentaremos comenzar con un proyecto simple el cual nos permitirá conocer, básicamente, el IDE y transferir un programa a la placa de demostración. Luego lo ejecutaremos en modo debugger para analizarlo paso a paso.
Este programa simplemente invertirá el estado de uno de los leds (port GPIOB1-pin 11) al presionar el pulsador IRQ (input IREQA-pin 16).
Partiendo de la figura 3, dentro de la ventana *Bean Selector* solapa *Categories* agregamos el *bean Button* al proyecto haciendo doble clic sobre el árbol *CPU External Devices->EVM->Button*. Este bean, a diferencia del *Switch*, permite asociarlo al pin IRQA del DSC. Para lograr esto hacer doble clic en el nombre del *bean* agregado en la ventana *Proyect Panel* solapa *Processor Expert* para que se abra el *Bean Inspector* en la sección central de la pantalla. En esta ventana se tienen todas las configuraciones posibles para el bean seleccionado junto con los eventos y métodos asociados (ver solapas) permitiendo habilitarlos o no.
En el renglón *Used pin* clickear sobre el signo mayor (>) para que se abra el *bean* heredado que nos facilita la comunicación con el botón. Abrir el combo *Pin* y seleccionar, de entre todas las opciones, la que indica *IRQA*. Esto nos permite vincular la IRQA con el *bean Button*. De esta manera ya tenemos configurado el botón para que, al presionarlo, genere el evento de interrupción y el programa salte a la función de atención del evento, BTN1_OnButton en este caso.
Para el agregado del LED, dentro de la ventana *Bean Selector* solapa *Categories* agregamos el *bean Led* al proyecto haciendo doble clic sobre el árbol *CPU External Devices->EVM->Led*.
Una vez agregado, al igual que para el botón, seleccionamos el signo mayor dentro de *Output Pin* en el *bean inspector*. Esto nos redirige al *bean heredado* dentro del cual podemos alterar el pin que vincularemos al LED. En nuestro caso deberemos reemplazarlo por GPIOB1_RXD0.
Para evitar la generación de código innecesario deshabilitaremos todos los módulos, en el *Proyect Panel*, excepto *Toggle* que será el que utilizaremos para alterar el estado del led. Para lograr esto simplemente se hace doble clic sobre el método a deshabilitar. Todos aquellos que tengan ☒ indica que están deshabilitados y los que tengan ☑ indica que están habilitados.
Ya estamos en condiciones de realizar la primer compilación del proyecto para verificar que no existan errores y para analizar los archivos que genera el PE. Para ello presionamos F7. Si no existieron errores la ventana *Proyect Panel* deberá quedar como sigue:

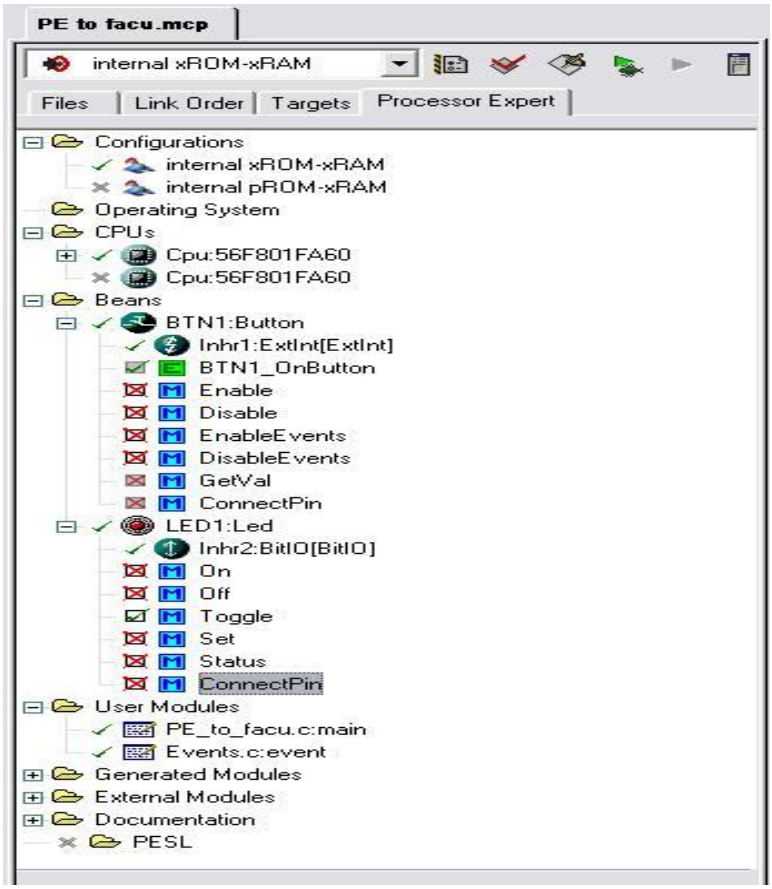

Figura 4

Dentro de *User Modules* se generaron dos archivos que son los que analizaremos y en los cuales agregaremos nuestras modificaciones.